Chapter 3

Application à l'Endoscopie Virtuelle et à Plusieurs Problèmes en Imagerie Médicale

Abstract — Dans la première partie de ce chapitre on s'intéresse à la construction d'une méthode d'extraction automatique de chemins pour l'Endoscopie Virtuelle. C'est un procédé de navigation dans des images 3D, qui nécessite la définition d'une trajectoire précise pour l'observation en image de synthèse de l'intérieur du corps humain. Nous avons appliqué notre méthode d'extraction de chemins minimaux du chapitre 2 pour contruire ces trajectoires rapidement et de manière la plus automatique possible.
La seconde application concerne, au contraire, la construction de chemins de manière interactive, permettant à un utilisateur de dessiner rapidement les contours d'un objet, en ne précisant qu'un point de départ, sur le modèle des techniques appelées *Live-Wire* de *Falcao et Udupa* [49] ainsi que de *Mortensen et Barrett* [127]. Sur la base de notre méthode, nous avons développé un outil similaire, incluant une possibilité d'adapter les paramètres du modèle, au cours de la segmentation, en l'entrainant à reconnaitre les contours qui nous intéressent.

Abstract — First section concerns the creation of a fully automatic path tracker for *Virtual Endoscopy* . *Virtual Endoscopy* visualizes the inner surfaces of structures present in volumetric data in 3D images. As navigation through the inner structures quickly becomes a complicated procedure, especially when these structures are strongly curved, often a trajectory through the structure is used. We applied our path extraction technique developed in chapter 2 in order to build an accurate and fast tool to automatically builds those trajectories.
For the second application, we have focused on the need in many applications, as in medical imaging, for interactive segmentation, offering the possibility to a non-expert to draw quickly the boundary of an object, following *Live-Wire* technique of *Falcao and Udupa* [49], and *Mortensen and Barrett* [127]. *Live-Wire* methods restrict the intervention of the user to the setting of a start point in an image. Using our path extraction algorithms, we have developed the same tool, including a training method that adapt the different parameters of the model *On-The-Fly* .

3.1 *Virtual Endoscopy*

Visualization of volumetric medical image data plays a crucial part for diagnosis and therapy planning. The better the anatomy and the pathology are understood, the more efficiently one can operate with low risk. Various post-processing techniques have been available, to enable the radiologist to recognize a pathological condition, in the shortest amount of time: three 2D orthogonal views (see figure 3.1), maximum intensity projection (MIP, and its variants, see figure 3.2), surface and volume rendering.

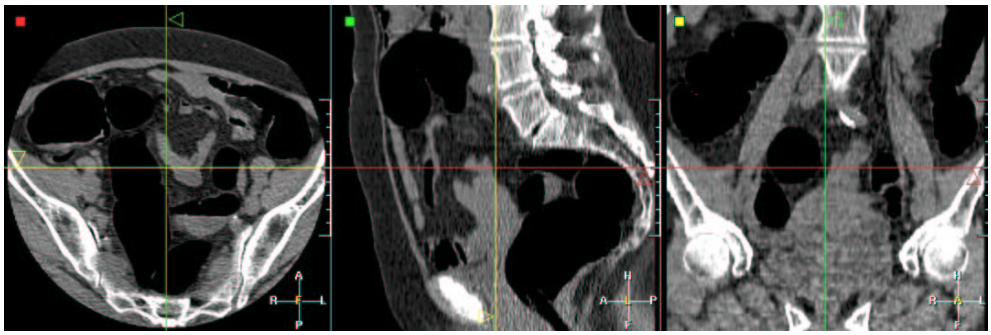


Figure 3.1. 3D CT scanner of the colon: Display of the dataset of $512 \times 512 \times 121$ voxels using three orthogonal views; air was injected in the colon (dark regions of the image) before acquisition in order to inflate the object and increase accuracy of the reconstruction of the colon.

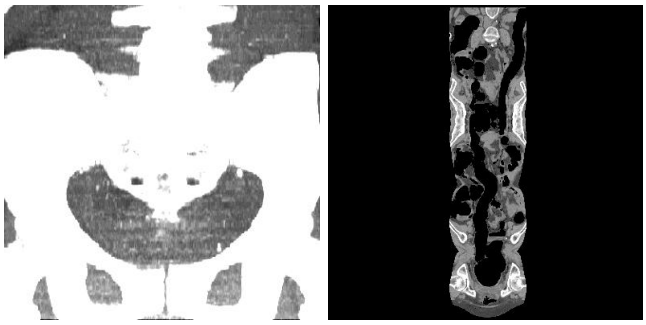


Figure 3.2. Different rendering techniques of a 3D CT scanner of the colon: left image is a Maximum intensity projection **MIP** and right image is a curved reformat image of the dataset shown in figure 3.1.

The maximum intensity projection requires to chose a direction of projection, therefore the volume is projected on a 2D plane, keeping only the brightest image points (in figure 3.2-left, the brightest intensities are given by the bones).

The curved reformat image in figure 3.2-right is a particular case of a multi-planar reformat image **MPR** : A **MPR** is a cross-sectional image of the 3D volume, along

an arbitrary plane, and a curvilinear reformat is a section along a surface generated by a bi-dimensional curve $f : \mathbb{R} \rightarrow \mathbb{R}, \mathrm{y} = \mathrm{f(x)}, \forall \mathrm{z} \in \mathbb{R}$. For a 3D trajectory, there are 3 different ways to represent the **MPR** , by fixing either x, y or z. The path is placed in the target of investigation (the colon in figure 3.1), and the organ appears in the reconstruction in its full length. Advantage is to display the whole complex shape of the colon on a single 2D image, but the anatomical objects are distorted, and dimensions are not reliable on the final image.

Virtual Endoscopy allows by means of surface/volume rendering techniques to visually inspect regions of the body that are dangerous and/or impossible to reach physically with a camera (e.g behind an airway stenosis or obstruction, or too small). An extensive definition of *Virtual Endoscopy* can be found in [80, 146]. In chapter 7, we detail visualization techniques based on volume and surface renderings.

Virtual Endoscopy techniques can be divided into two groups of methods that can collaborate:

- techniques which deal with simulation of a real endoscope motion; In this case, *Virtual Endoscopy* is very interactive, simulating the motion of a camera inside the body, based on an extracted anatomical object that is modeled using rigid body dynamics; good examples of this simulation are presented in [77, 131].

- techniques which focus on the observation of the interior of anatomical objects by extracting trajectories inside them, see *Yeorong et al.* [192] for an example.

We have focused on the second kind of techniques. However, the minimal path techniques can also be useful for the first kind of methods: Kimmel and Sethian have applied the Fast-Marching algorithm for a robotic application in [163], for the motion of an object with a certain shape and orientation in an image with obstacles. We have also investigate this field in section 2.4, modeling the movement of an object, discretizing *Eikonal equation* in a space that describes the object position and orientation. But adding a dimension to the problem could lead to huge computing costs for an interactive 3D application.

Figure 3.3 is the usual framework that builds a *Virtual Endoscopy* facility, and is usually composed of two parts:

- A Path construction part, which provides the successive locations of the fly-through in the tubular structure of interest (see figure 3.3-left);

- Three dimensional interior viewing along the endoscopic path. Those views are adjoined creating an animation which simulates a virtual fly-through through them (see figure 3.3-right).

3.1.1 Medical relevance

In recent years, computerized post-processing techniques of image data from cross-sectional imaging modalities has received increasing recognition in the field of medicine. Technical developments of acquisition systems such as CT and MRI have improved along with continuously increasing spatial resolution. But if each axial image were

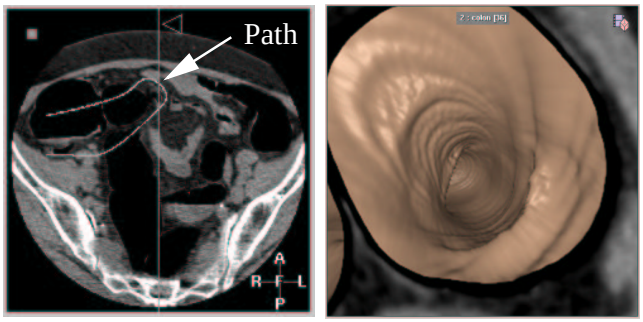


Figure 3.3. *Virtual Endoscopy* framework: Left image shows an example of trajectory manually drawn using the orthoviewer facility; right image is a volume rendering of the dataset, using a ramp function manually parameterized, at a position along this path.

to be viewed before one could proceed to the next image, the viewer would require an enormous amount of time to shift through all slices. At the same time, growing computer performance has created the opportunity to display anatomical structures in a comprehensible manner. *Virtual Endoscopy* is one of the most recent innovations in the spectrum of post-processing techniques. The predominant motive is to simulate conventional endoscopy, as in figure 3.4-left, by means of a non-invasive and safe technique, presenting the image data included in the original slices, in a movie-mode, in such a fashion that the radiologist is able to differentiate between that which is healthy and that which is pathological. Figure 3.4 shows how a clinician is able to detect pathologies using the volume rendering tool associated to the centered path extraction. But the true diagnostic performance of *Virtual Endoscopy* and the ability

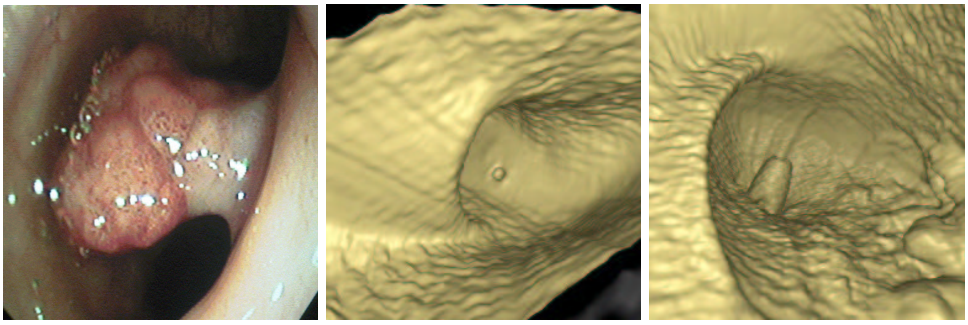


Figure 3.4. Pathologies detected with the volume rendering: Left image is a conventional endoscopy image of the colon; the middle image displays a small cavity which is a diverticulum in the colon surface; on the lower-right part of the right image, the bumps are polyps.

to reproduce acceptable results outside of research protocol has not been established yet. Clinical validation is still an on-going process, and researchers focus on the ap-

plication and validation of this technique for particular organs and pathologies. For colorectal cancer (CRC) (see figure 3.4-right), studies (see [146]) show that net effect of the use of *Virtual Endoscopy* could reduce cancer-related deaths in the future.

3.1.2 Problem with the path construction

A major drawback in general remains when the path construction is left to the user who manually has to "guide" the virtual endoscope/camera. The required interactivity can be very tedious for complex structures such as the colon for example. Actually, on most clinical platforms the user must define all path points manually, using for example three 2D orthogonal views, as shown in figure 3.1, leading to problems as the following:

- Since the anatomical objects have often complex shapes, they tend to pass in and out of the three orthogonal planes. Consequently the right location is accomplished by successively entering the projection of the desired point in each of the three planes;
- The path is approximated between the user defined points by lines or Bezier splines. If the number of points is not sufficient, it can easily cross an anatomical wall.

Path construction in 3D images is thus a very critical task and precise anatomical knowledge of the structure is needed to set a suitable trajectory, with the minimum required interactivity.

Numerous techniques try to automate this path construction process. Most of them use a skeletonization technique, like in [192], in order to extract a centerline in the dataset. But extracting the skeletons of an anatomical shape requires first to segment it. And the skeleton often consists in lots of discontinuous trajectories, and post-processing, as done by *Tek and Kimia* [173] is necessary to isolate and smooth the final path. The front propagation techniques studied in this application in contrast to other methods does not require any pre- or post-processing as explained in section 2.3.

It is sometimes necessary to smooth the path extracted by the front propagation. The point of view in the volume rendering of the tubular structure is very important, because it constrains the result of the examination. Thus, during the virtual fly through, the point of view of the camera must change smoothly. Traditionally, the position of the virtual camera frame at a particular path point is orthogonal to the path. If the path is not smooth, the point of view of the virtual camera will change in an abrupt manner. There are two ways to achieve this regularization:

- by modifying the view angle of the virtual camera, being no more orthogonal to the path, but looking in the direction of a path point which is located far from its current position, or using a running average of the local direction of the camera;
- by increasing the weight w in equation (1.3) since it has a smoothing effect on the minimal path (see appendix for details). We preferred to use this technique in the following examples, since it is efficient and very simple to add.

3.1.3 Proposed solution for colonoscopy

The method detailed here is illustrated by results performed on the volumetric CT scan shown in figure 3.1.

Building a potential for virtual colonoscopy

The target is to build a potential P with the 3D data set allowing paths to stay inside the anatomical shapes where end points are located. We thus define the potential by a general model $\tilde{P}(x) = |I(x) - I_{mean}|^{\alpha} + w$.

First, the potential must be lower inside the colon in order to propagate the front faster, and to avoid problems with crossing the edges of the anatomical object. In a colon CT scan, an average position I_{mean} of the colon grey level in the histogram can be defined (see figure 3.5) as a peak in the histogram where $I_{mean} = 200$. Secondly,

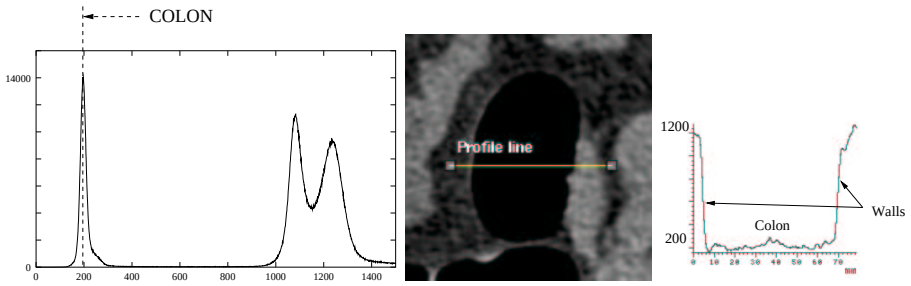


Figure 3.5. Global and Local study of the image intensities: Left image is the histogram of the whole 3D dataset where the colon can be clearly distinguished; middle image is a slice of the colon where a profile line was drawn in order to observe the variation of intensities that are displayed in right image.

if the path to be extracted is very long, the situation can lead to pathological cases, and the front can go through potential *walls*. This is frequent for large objects that have complex shapes and very thin edges, as colon. Then, edges should be enhanced to enable long trajectories, with a non-linear function. We thus take $\alpha = 2$ in order to enhance the dynamic of the image with a quadratic function.

However, this potential does not produce paths relevant for *Virtual Endoscopy* . Indeed, paths should remain not only in the anatomical object of interest but as far as possible from its edges. In order to achieve this target, we use the centering potential method as detailed in section 2.3. We first need to obtain a *shape* information. In fact, a CT scan of the colon contains already a shape information sufficient to constrain a front propagation. In figure 3.5-middle is shown a slice of a colon volumetric data set, and figure 3.5-right shows the corresponding grey level profile. Air fills the colon and is represented in our CT image by a grey level around 200 (see figure 3.5-right), while edges are defined by a grey intensity around 1200. Then, using the potential $\tilde{P}(x) = |I(x) - I_{mean}|^{\alpha} + w$, the front obtained through *Fast Marching* is stopped by the anatomical shapes. Figure 3.6 shows the propagation of the wave equation, according to the penalty defined previously. It also illustrates the fact that the *Fast-Marching* can act also as a segmentation tool.

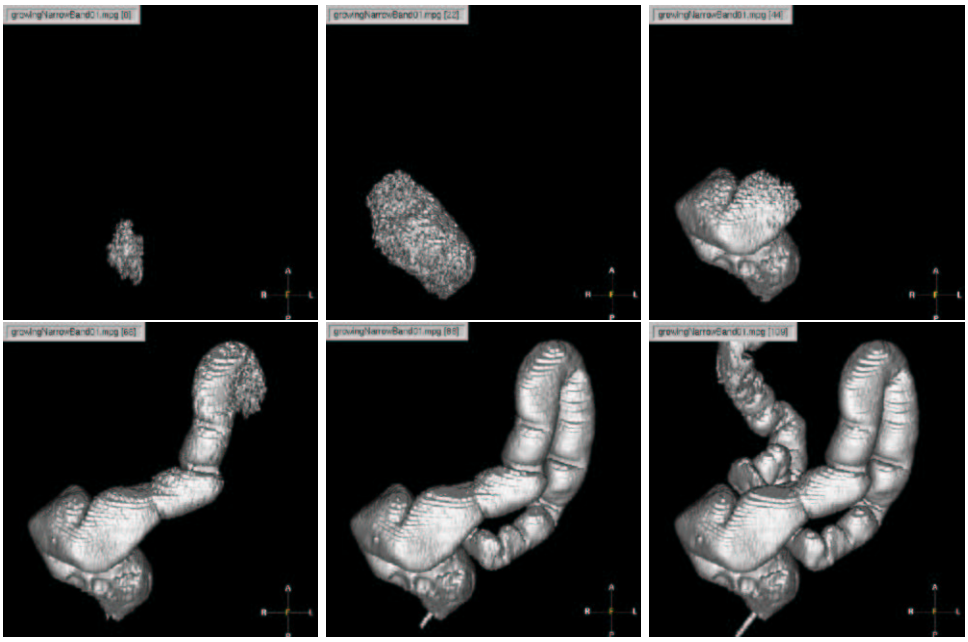


Figure 3.6. Wave propagation inside the colon dataset: these are volume renderings of the *Alive* points at different consecutive iterations during propagation.

Path centering technique

The edges are obtained via a first propagation: in figure 3.6 we can see the evolution of the *narrow band* during propagation. It gives a rough segmentation of the colon and provides a good information and a fast re-initialization technique to compute the distance to the edges. Using this distance map as a potential (from equation (2.7)) that indicates the distance to the walls, we can correct the initial path as shown in figure 3.7-left: the new path remains more in the middle of the colon. And the value of the parameter d can be derived from anatomical characteristics. If we know approximately the section of the colon along the path we can easily choose a value to stay in the center of the tubular structure.

The two different figures 3.7-middle and 3.7-right display the view of the interior of the colon from both paths shown in figure 3.7-left. With the initial potential, the path is near the wall, and we see the u-turn, whereas with the new path, the view is centered into the colon, giving a more correct view of the inside of the colon. The new centered path is smooth because this final propagation is done on a synthetic potential (the distance to the walls) where noise has been removed.

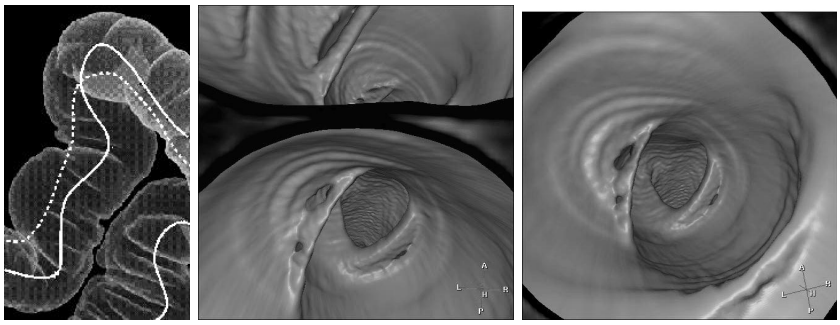


Figure 3.7. Comparing normal and centered paths: The left image is a Tissue Transition Projection (TTP) of the colon surface; middle image is the endoscopic view obtained with the classical trajectory (represented with a doted curve) in the U-turn shown in the left image; right image shows the endoscopic view resulting from the centered path (represented with a plain curve on left image) at the same position in the colon.

3.1.4 Results on objects filled with air

Results on Colonoscopy

Virtual colonoscopy is one of the most exciting developments in gastrointestinal radiology. It is a non-invasive, well-tolerated, and safe technique for evaluating colorectal cancer (CRC). **CRC** represents the third most frequently diagnosed cancer worldwide. For the United States, it is estimated that 129,000 new cases were diagnosed in 1999 [168]. Preliminary results indicate that the accuracy of virtual colonoscopy exceeds that of conventional endoscopy. Our automatic path extraction provides in a very fast process a path that remain inside the structure and avoid collisions with the wall, following in a smooth manner the centerline of the colon. Once this path is created, navigating is simply a matter of positioning the view along the path. Figure 3.8 shows the difference between the paths extracted with the classical method, and the centering method.

Notice that in figure 3.8-middle, the path has been smoothed by the centering method. This is mainly due to the fact that the noise inside the colon has been *eaten* by the first propagation. The small variations in intensity allows the front to propagate in all the colon, and considering the distance to the borders of the object *cleans* the anatomical object, by considering an artificial penalty $P(x, y, z) = 1\ \forall (x, y, z) \in \mathbb{N}^3 \cap \text{colon}$. Figure 3.8-right illustrates the complexity of the path extracted: intersecting the same slice several time, the user who wants to extract it manually must have strong skills in anatomy, and time. Figure 3.9 shows the corresponding endoscopic viewings generated with those two paths.

Results on the Trachea

The virtual inspection of the trachea is the same problem as in the colon. It is even simpler, due to the topology of the organ observed. Figure 3.10-left displays a 3D

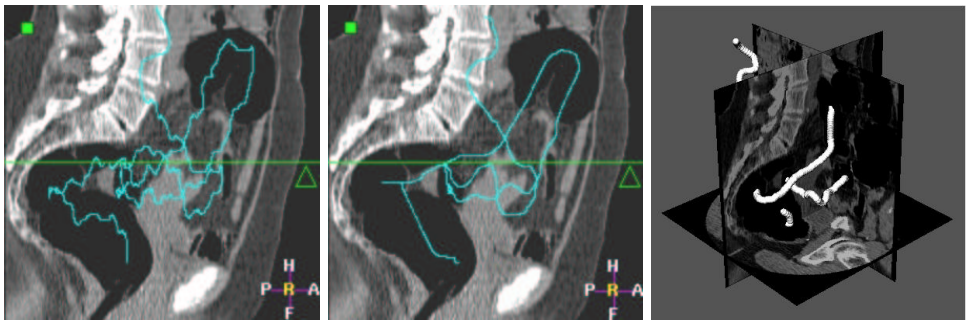


Figure 3.8. Path extraction in the 3D CT scanner of the colon: Left image is the superposition of a path extracted from only one point manually located inside the colon over a slice of the dataset; middle image displays the centered path in the same configuration; right image is the representation in 3D of the trajectory displayed in middle image intersecting three planes, where the dataset has been mapped.

CT dataset of a trachea with three orthogonal views. Air fills the object and gives a shape information all along from throat to lungs. Therefore, the anatomical object having a very simple shape, the path construction with one or two fixed points is easier than in the colon case. One example path tracks the trachea, using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 200|^2 + 1$). This path has been used to display the **MPR** view of figure 3.10-right. An endoscopic view along the same path is displayed in figure 3.11. The inspection of the trachea is often part of the inspection of the whole tracheobronchial tree. In part III, we will study more precisely the tracheobronchial tree. The examination of the complete tree needs new algorithms to extract the complete tree structures, and to segment the borders of the bronchi. This implies use of more complicated algorithms than *Fast-Marching* and will be further developed.

3.1.5 Results on Arteries and vessels

Is endoscopy a useful tool for artery and vessel examination?

The volume-based rendering tool use an opacity threshold which can create severe artifacts on the endoscopic viewings. And the quality of the images is a direct result of the homogeneous opacification of the blood. The blood is made visible using contrast products, and this image information depends now on the section of the object. It can be also severely impaired in areas of turbulent flows.

Review of the axial slices alone is somehow sufficient for the diagnosis assessment of several pathologies. But, still automatic path extraction provides important visualization assessments on the datasets. It can be used with multi planar reformatting **MPR** images, which enable to see the tubular objects with optimal slice orientation, depicting spatial relationships between the object and its pathologies. But *Virtual Endoscopy* offers an intraluminal view of the arteries and veins and allows inspection of pathologies, like stenosis. It enhances visualization, by clearly showing the spatial

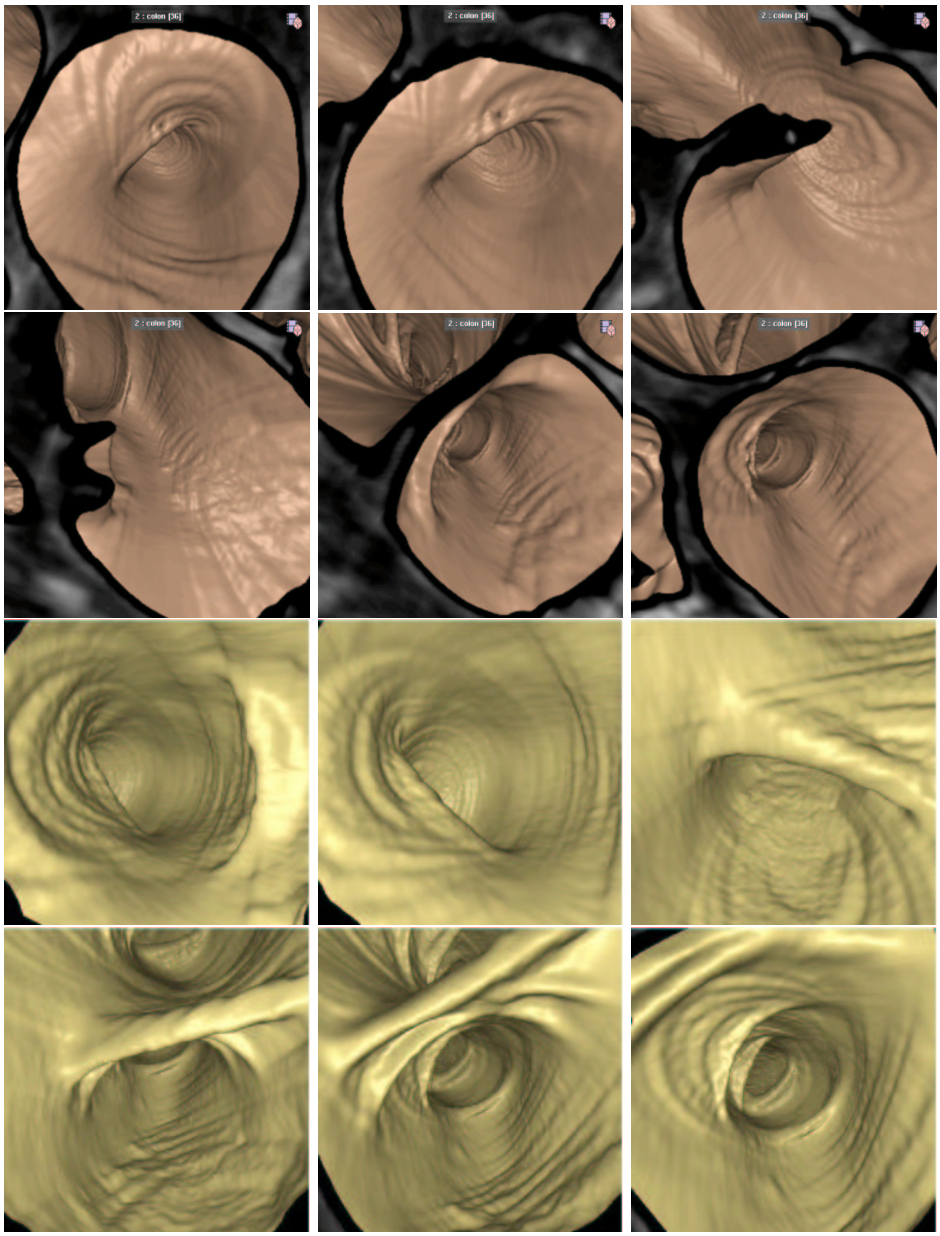


Figure 3.9. *Virtual Endoscopy* in the colon, minimal and centered paths: On the first two rows the trajectory being the shortest, the intersection plan of the camera shows adjacent structures and shrinks the validity of the examination; on the two last rows, the trajectory being centered, the point of view of the virtual camera shows the entire tube, and the validity of the examination is increased.

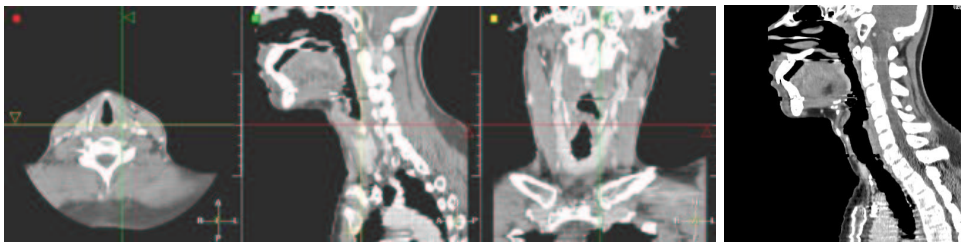


Figure 3.10. 3D CT scanner of the trachea: Left image is a dataset of $320 \times 320 \times 234$ voxels; the interior of the trachea is very easy to characterize from the whole image since it is always filled with air; right image is a **MPR** image along the trajectory extracted.

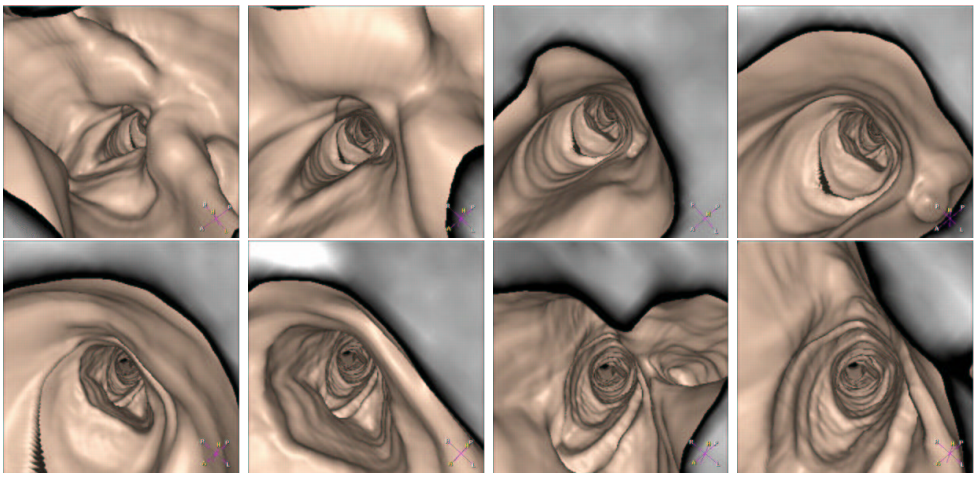


Figure 3.11. *Virtual Endoscopy* in the Trachea: starting from the mouth, the virtual camera goes straight to the bifurcations between the lungs, following the path extracted (see **MPR** view in figure 3.10-right).

relationship between the pathology and the object, comparing those pathologies before and after treatment, as done for stent placement in abdominal aortic aneurysm **AAA** . It can be also input in systems for the virtual planning of endoluminal aortic stents (see [186]).

This gives justification for the following study on the automatic path extraction in veins and arteries. But the grey-level information is more complex in the case the signal is obtained through injection of a dye product in the object of interest. The boundaries of the object are more difficult to extract, therefore the path centering technique developed in section 2.3 and previously applied to the colon case, is no longer valid for the following. However, automatic path extraction has remained valid, despite the variability of the new penalty information. Centering techniques for those kind of objects involve the use of more complex algorithms, like achieved segmentation algorithms (see *McInerney and Terzopoulos* [115] for a review of med-

ical image segmentation with deformable models), and have been later developed *a posteriori* in part III of the thesis.

3D MR image of the Aorta

A test was made on an aorta MR dataset. Figure 3.12-left displays this 3D MR dataset of the abdominal aorta using three orthogonal views. Contrast product was injected

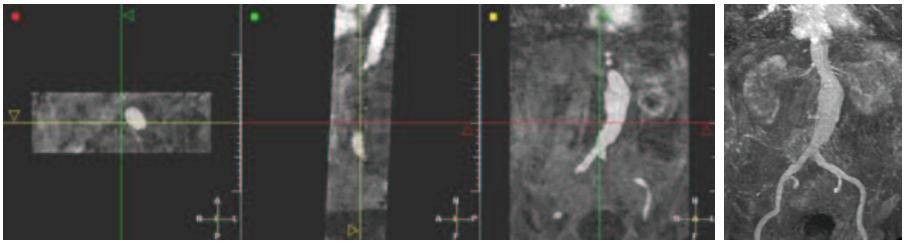


Figure 3.12. 3D MR image of the aorta: Dataset of $256 \times 256 \times 60$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the right image is a threshold based volume rendering of the aorta itself, which highlights that the anatomical object has a visible pathology: an Abdominal Aortic Aneurysm (AAA).

before acquisition. On the **MIP** view, in figure 3.12-right, the important variation of the section of the object, upon the bifurcation of the iliac arteries, clearly indicates an an Abdominal Aortic Aneurysm **AAA** . The dye fills the aorta, and makes it visible, among other soft tissues, while bones are not rendered. The propagation measure is based on a nonlinear function of the intensity of the contrast solution that fills the aorta. This data set is difficult since the intensity of the contrast product will vary along the aorta (the contrast bolus dilutes during the acquisition time). Due to this non-uniformity, paths can cross other anatomical structures with similar intensities if the mean value inside the aorta is not set correctly by the user. Our example path tracks one iliac artery (see figure 3.13), using the potential $\tilde{P}(x) = |I(x) - 1000|^2 + 10$ in the MR scan. The dataset contains noise, and we must use an important weight to smooth the extracted paths. We have displayed a sample of the endoscopic views of the aorta along the path in figure 3.14. During the virtual fly through, the observer clearly notice the important variation of the object cross-section. The movie in figure 3.14 depicts this pathology, even if the threshold-based volume rendering produces artifacts, when using the image grey level information obtained from a contrast enhanced MR image.

3D CT Scanner of the Aorta

Figure 3.15 displays a 3D CT dataset of the abdominal aorta using three orthogonal views. The contrast product injected before acquisition fills the aorta, and make it visible among other objects. The problem is similar to path extraction in a MR image, as done previously. Main difference here lies in the high resolution of the CT scanner

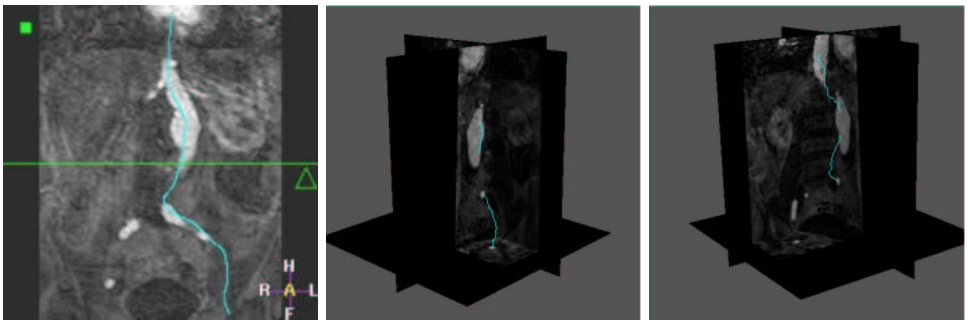


Figure 3.13. Path extraction in the 3D MR image of the aorta: Left image is the superposition of one path extracted between two points manually located inside the aorta over a slice of the dataset; middle image is a curved reformat sagittal image along one of the paths which tracks the right iliac artery; right image is a volume rendering view based on an opacity function parameterized manually along the same trajectory.

dataset of figure 3.15, towards the MRA dataset of figure 3.12. This high resolution increases significantly the computing time of the method, but even if the dynamic of the images is very different, it does not lead to major differences in the parameterization of the path extraction process. Figure 3.16-left displays several paths extracted with the same seed point. Figure 3.16-middle is a curved reformat view along one of the trajectories extracted. This kind of view enables to validate trajectories by verifying that the section drawn always intersect the structure of interest.

3D MR image of the brain vessels

Tests were performed on brain vessels in a MRA scan. Three orthogonal slices of this dataset are shown in figure 3.17 together with a path extracted. The problem is different, because there is only signal from the dye in the cerebral blood vessels. All other structures have been removed. The main difficulty here lies in the variations of the dye intensity. The example path tracks the superior sagittal venous canal (the vein on the top of the head, as shown in figure 3.17-right), using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 100|^2 + 1$). Two views of the extracted path in 3D are displayed in figure 3.18 together with 3 orthogonal slices of the dataset. A sample of the virtual fly-through along the brain vessel is displayed in figure 3.19.

3.1.6 Clinical study

Goal

A multi-user clinical study was performed using a prototype based on EasyVision (Philips Medical Systems). The purpose was to measure the speed and user-dependence of the automatic path tracker. To this aim, the path tracking tool has been evaluated by 5 different operators :

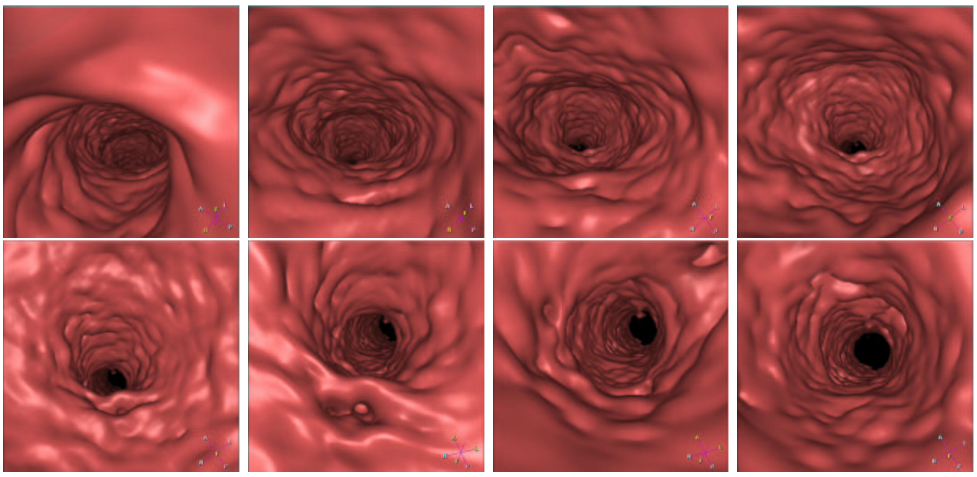


Figure 3.14. *Virtual Endoscopy* in the Aorta: starting from one iliac artery, the virtual camera goes to the top of the aorta, and the section of the object becomes larger when it is located inside the aneurysm.

- two physicians with manual path tracking experience ($P1$, $P2$);
- two operators with abdominal anatomy knowledge but no virtual colonoscopy practice ($M1$, $M2$);
- one reference operator (R) familiar with the automatic path tracking tool.

As a comparison the user R also manually defined a path twice on the same dataset.

Data

Spiral CT data (5 mm slice thickness, 3 mm reconstruction interval) from 15 patients were used, corresponding to a total of 29 scans. During the patient preparation phase the colon was emptied as much as possible, and distended by inflating room air. In most cases, both prone and supine scanning was done.

Measurements

Time For each user, the wall clock time was measured using an automatic logging mechanism. In general, path construction consisted of following steps :

1. load and inspect data
2. place starting point in the cecum
3. track + center path
4. check result and modify/continue tracking if necessary

The time necessary to perform steps 3 and 4 were measured and both user interaction time and calculation time were taken into account.

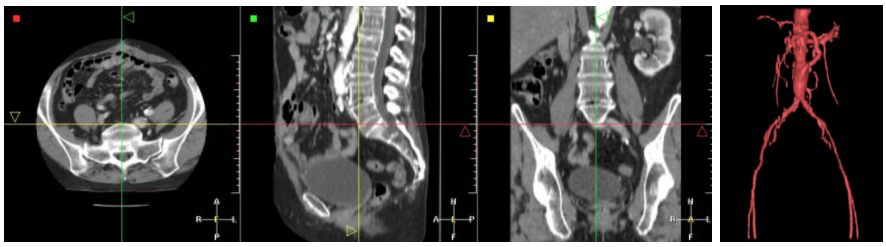


Figure 3.15. 3D CT Scanner of the aorta: Dataset of $512 \times 512 \times 173$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the anatomical object does not have a visible pathology; the right image is a threshold based volume rendering of the aorta itself (the **MIP** view is disturbed by the intensities of the bones).

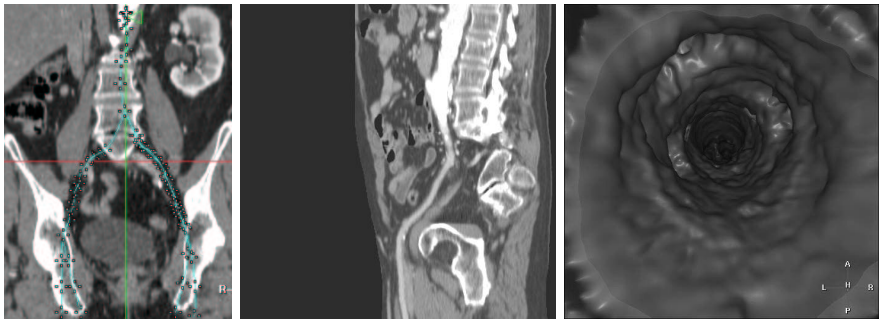


Figure 3.16. Path extraction in the 3D CT scanner of the aorta: Left image is the superposition of several paths extracted from the same seed point at the top of the aorta, over one slice of the dataset; middle image is a curved reformat view along one of the trajectories extracted; right image is an endoscopic view along this path.

User-dependence To measure the user-dependence of the path tracker, resulting paths P from different users were compared to the corresponding path R obtained by reference user in the following way :

1. **Warp path $P(i)$ and $R(j)$ to a common length parameter k**
   We want to locally stretch and compress paths $P(i)$ and $R(j)$ using warping functions $w_P(k)$ and $w_R(k)$. The goal is to obtain the optimal warping satisfying

$$(w_P, w_R) = \arg \min_{w_1, w_2} \left(\sum_k \|P(w_1(k)) - R(w_2(k))\| \right) \tag{3.1}$$

   where all points of P and R are addressed by the warping. This mapping is calculated using the Dynamic Time Warp algorithm [145].

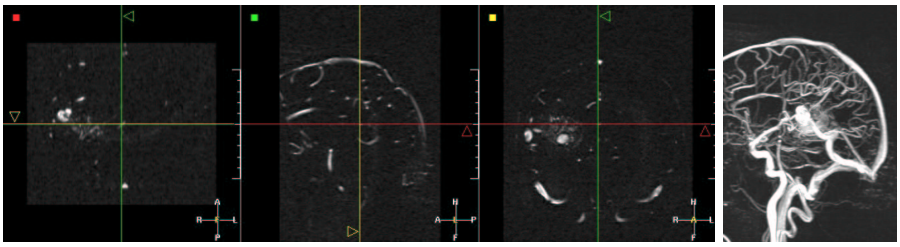


Figure 3.17. 3D MR Angiography image of the brain vessels: Dataset of $256 \times 256 \times 150$ voxels; It is obtained by subtracting two acquisition: one before, and one after injection of a dye product; thus there is only signal coming from moving objects; right image is a **MIP** view of this dataset.

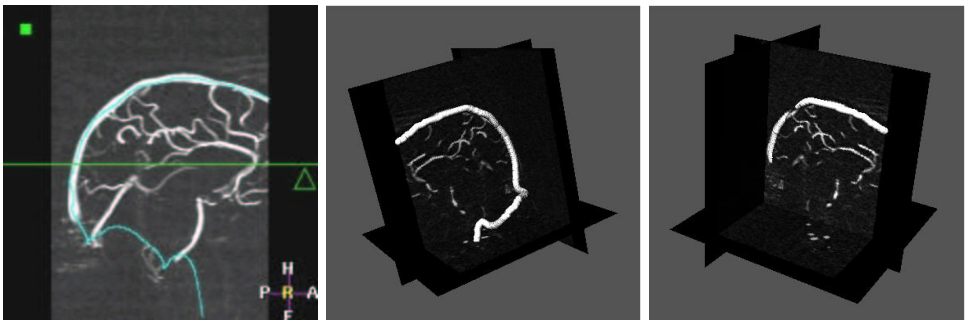


Figure 3.18. Path extraction in the 3D MR Angiography image of the brain vessels: Left image is the superposition of one path extracted between two points manually located inside the superior sagittal sinus over a slice of the dataset; middle and right images are the representation in 3D of this trajectory intersecting three planes, where the dataset has been mapped.

2. **Calculate Euclidean distances d between corresponding points**
   After warping, the path distances d can be calculated as

$$d_{P,R}(k) = \|P(w_P(k)) - R(w_R(k))\| \tag{3.2}$$

   and are shown in Fig.3.20 as a function of the common path length k.

3. **Extract the common part**
   To exclude the effect of the exact start and end point, we only consider the common part of paths P and R, which is defined in Fig.3.20.

4. **Measure the similarity $\mathcal{S}$ between the paths**
   We propose to measure the similarity by using the percentage of the common path length k where the distance d is below a threshold t, written as $\mathcal{S}^t_{(P,R)}$.

The chosen measure for path similarity $\mathcal{S}^t_{(P,R)}$ is symmetrical, it is robust to a complete different choice in starting position, and it gives a more reliable result of

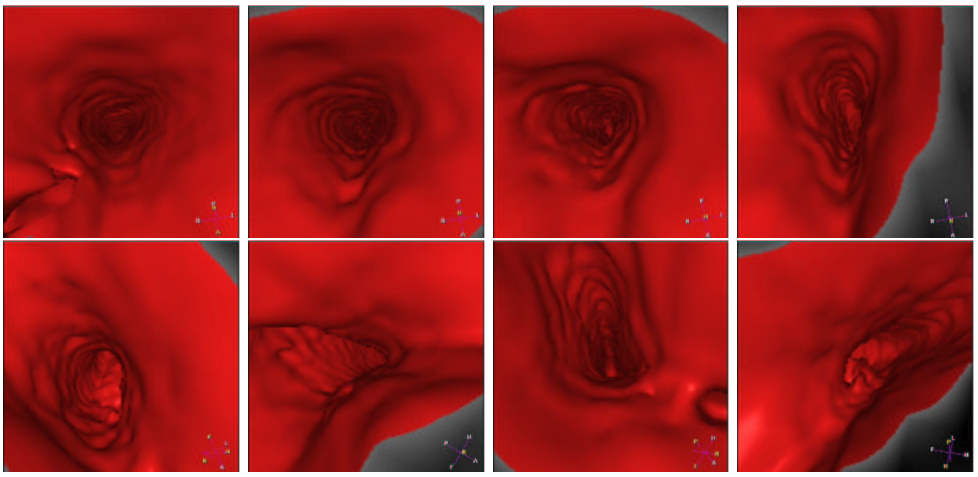


Figure 3.19. *Virtual Endoscopy* in a brain vessel: the virtual camera goes into the superior sagittal sinus venous canal.

similarity between paths than e.g. the maximum distance.

Results

Time The results of the time measurements are shown in Table 3.1. The average time needed for path tracking is 4.8 minutes per scan, measuring both user interaction time and calculation time.

time	P1	P2	M1	M2	R	average	manual
user time	2.8	4.6	5.3	4.5	2.0	3.6	30.0
calculation time	1.0	1.0	1.2	1.7	1.3	1.2	
total time	3.8	5.6	6.5	6.2	3.2	**4.8**	30.0

Table 3.1. Timing results of the automatic path tracker:Time expressed in minutes per scan.

No significant differences were found between the experienced physicians ($P1$ and $P2$) and the other users ($M1$, $M2$), excluding the reference user R. These results are in agreement with a previously published study [147], where an average of 4.5 minutes was measured on 27 cases. The timing for the manual case has no statistical meaning, but is merely given for comparison.

User-dependence Table 3.2 summarizes the measurements of the path correspondence using the similarity measures $\mathcal{S}^2$ (2 mm threshold) and $\mathcal{S}^5$ (5 mm threshold) as defined in section 3.1.6.

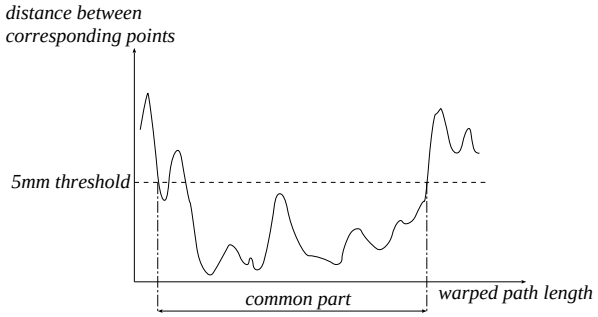


Figure 3.20. The Euclidean difference between corresponding points plotted versus the warped path length k. The common part is defined as the longest possible stretch between two positions where the distance falls below the threshold of 5 mm.

correspondence	P1	P2	M1	M2	average	manual
$\mathcal{S}^2$	79%	89%	81%	92%	85%	20%
$\mathcal{S}^5$	89%	97%	93%	95%	**94**%	68%

Table 3.2. Correspondence with reference path: Correspondence expressed in percentage of the path length where the reference path is closer than 2 mm (first row) or 5 mm (second row).

On the average, an automatically defined path differs less than 5 mm from the reference over 94% of its length. Again, the results of the manual path tracking is given for comparison. Both manually tracked paths differed less than 5 mm over only 68% of their lengths.

Automatic path tracking provides a fast and easy way to determine the colon centerline. The resulting path lies completely inside the colon, is as much centered as possible and is smooth.

The path tracker can be used by less experienced operators without significant differences in time or resulting centerline. This is an important result, since it allows to separate the path tracking task from the actual inspection task. The former task can be done as a preprocessing step by a different person since the results of the path tracker are largely operator-independent. Separating path tracking and inspection will increase the physician's efficacy, reduce the cost and allow a more widespread application of path-based navigation and visualization.

3.1.7 Last developments and perspectives

Figure 3.21 shows how the colon surface is unfolded using the point of view of the virtual camera which is given by the centered path extracted. This unfolding[1] enables to clearly see at each path position desired the rendering on each side of the camera

[1]This image was provided by Roel Truyen, from MIMIT Advanced Development, Philips Medical Systems, Best, Netherlands.

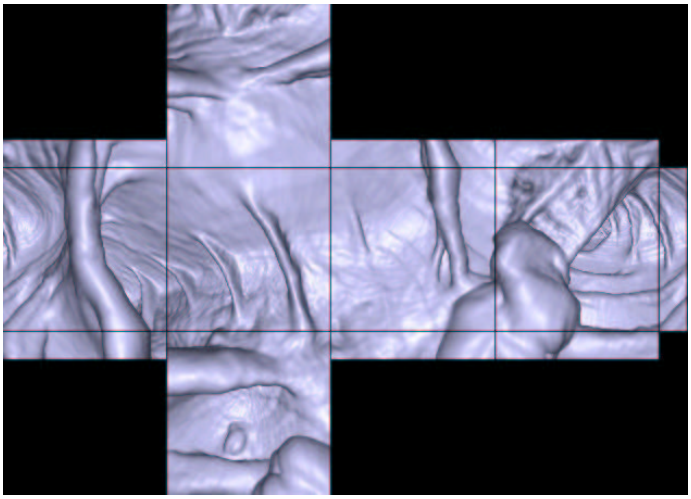


Figure 3.21. Unfolding the colon surface:.

and behind, thus increasing the accuracy of the diagnosis. For example, in the lower part of figure 3.21, the bump on the colon surface is a polyp that would not be detected if the point of view given follows the trajectory direction. Therefore, the only requirement for an accurate fly-through is to provide a trajectory that follow the centerline of the structure, as closely as possible, as this provides the best visualization of the surrounding walls. The path should also be smooth without unnecessary kinks.

Further work on the subject of endoscopy could be to simulate directly the trajectory of an endoscope inside the anatomical objects. The introduction of the angle as a dimension in *Eikonal equation* in section 2.4, and the algorithmic tricks developed to compute a minimal action for a moving object in the same section, could lead to the extraction of minimal paths in the human body for moving objects with a given shape. A straightforward application of this angular propagation for an object trajectory is to guide objects in a virtual endoscopic process. If the object is not a point, the modification of its orientation will now represent a cost to minimize. If this cost is also related to the refraction indices of the medium, the constraint will regularize not the trajectory of the object, but its orientation. For a virtual endoscopic camera, regularizing the point of view will lead to a better understanding of the scene and of the pathologies. Of course, the computing time is multiplied by the number of discretized angle in $[0; 2\pi]$, thus the problem is now four dimensional, and the path extraction is really time consuming. But we can forecast that the growing computer performance in the next years will make this improvement feasible.

3.2 Live-Wire

Approaches in image segmentation are numerous, ranging from fully automatic methods to fully manual methods. The first ones totally avoid user's interaction but still are

an unsolved problem: even if they are well adapted to specific cases their success can not be guaranteed in more general cases. The second ones are time-consuming, unrepeatable and inaccurate. To overcome these problems, interactive (or semi-automatic) methods are used. They combine knowledge of the user and computer capabilities to provide supervised segmentation, ranging from manual painting to minimal user intervention.

The target of this application was interactive and real-time extraction of features in medical images, independently from any acquisition modality. The aim was to develop a method to offer the possibility to a non-expert to draw quickly the boundary of an anatomical object. He could for example restrict its intervention to the deposition of a start point in an image. Then a contour had to be automatically found and drawn in real-time between this start point and the current cursor position. This contour should respond to a certain amount of constraints, such as internal and external forces and action of the user. The developed method should let the user validate or not the result. Taking this validation into account, the tool should be able to generate a kind of learning to better estimate the different parameters of the model.

In contour oriented segmentation, one approach is to define a boundary as the minimum of an energy function that comprises many components such as internal and external forces. In the literature, there exist many techniques to perform this minimization. First, classical active contours (also called Snakes or Deformable Boundaries), introduced by *Kass, Witkin and Terzopoulos* [82], have received a lot of attention during the past decade. But this technique presents three main problems. First, variational methods are very sensitive to the initialization step and often get trapped in a local minimum. Second, user's control cannot be applied during the extraction but only during the initialization (where the user has a whole contour to draw) and the validation stages of the segmentation. Third, the different parameters of the model are not meaningful enough in a user's viewpoint, especially for clinicians. The application of the minimal path theory to image segmentation is a more recent technique. With this approach the image is defined as an oriented graph characterized by its cost function. The boundary segmentation problem becomes an optimal path search problem between two nodes in the graph. This approach overcomes the problem of local minima by using either dynamic programming (*Dijkstra* [43]), or a front propagation equation (*Cohen and Kimmel* [34]), mapping the non-convex cost function into a convex function. Dynamic programming has also been used for classical snakes [4], but the proposed method is applied there to find the local deformation from an initial curve that gives the best energy descent. *Falcao and Udupa* with their *Live-Wire* [48, 49] and *Mortensen and Barrett* with their *Intelligent Scissors* [129, 126, 127, 128] introduce interactivity into the optimal path approach. Their method is based on Dijkstra's graph search algorithm and gives to the user a large control over the segmentation process. The idea is the following: a start point is selected by the user on the boundary to be extracted, and an optimal path is computed and drawn in real time between this start point and the current cursor position (see figure 3.22). Thus, user's control is applied also during the extraction. *Mortensen and Barrett* [10, 127] also introduce facilities called *Path-Cooling* and *On-The-Fly* training, which respectively lead to the possibility of drawing a closed contour, and to partial adaptation of the graph cost function. This technique seems to be the most

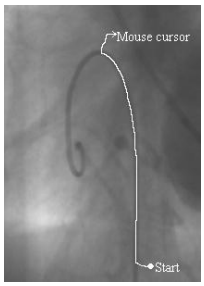


Figure 3.22. Principle of interactive contour extraction with optimal path method: the optimal trajectory is extracted in real time between a user defined starting point and the mouse cursor.

adequate according to our target. The application consisted therefore, first in the implementation of a method based on minimal path search inspired from *Live-Wire* and *Intelligent Scissors* tools, and second in using this implementation to go further with the idea of interaction and learning. The optimal path approach is developed in the first part of this section through the description of *Live-Wire* [49] and *Intelligent Scissors* [127]. The second part explains the adaptation we made of these techniques, including adjustments and improvements.

3.2.1 Existing methods

The aim of this work is interactive segmentation of contours in images. In contour oriented methods, one approach is to define the boundary as the minimum of an energy function, also called cost function. This cost function includes external energy terms describing the salient features that can be extracted from the image and internal energy terms ensuring the regularization of the segmented curve. With the minimal path approach the minimization is not global but local: the aim is to find the optimal boundary segment between two points, that is to say the contour segment where the energy is minimal. This can be achieved using the graph search theory, as detailed in section 1.3.1 .

Two different version of the minimal path extraction were used in the following:

1. first one is the *Dijkstra* algorithm [43], which is detailed in section 1.3.1.
2. second one is the *Fast-Marching* implementation presented in section 1.3.2.

Once the method is chosen, the result mainly depends on the choice for an acceptable cost function.

Cost function

The optimal-path search is guaranteed to find the solution of the minimization of the energy function between two points. But if this function does not pertain enough to the object to extract, the contour obtained by this method won't be right. That is

why defining a good and appropriate cost function is the essential task of this method. In both techniques, the processes are similar: the first step is the definition of some interesting features (F). A feature is supposed to describe certain properties of the boundary and of its environment (gray levels of the contour, of the background, ...). The next step is the conversion of these features into cost functions (C). A feature can for example be the gradient magnitude of the image, and the associated cost function can be the inverse of the gradient magnitude, giving higher cost to smaller gradients (i.e. weak contrasts) and lower cost to higher gradients (i.e. strong contrasts). The conversion of a feature into a cost function is achieved by a so called cost assignment function (CAF). Figure 3.23 illustrates all these concepts.

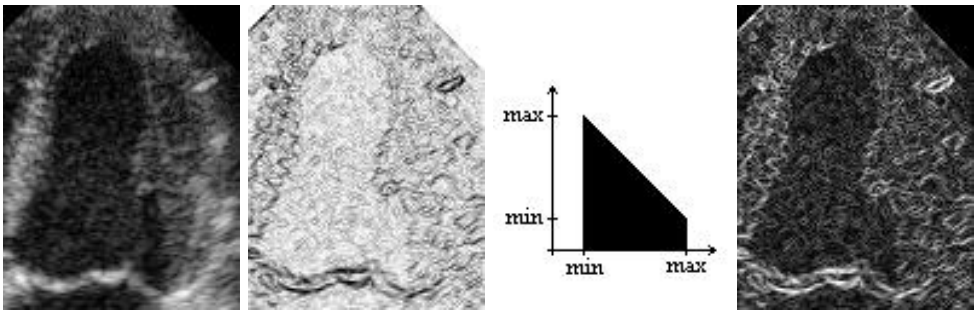


Figure 3.23. Illustration of the cost assignment function: From left to right: the initial image; the feature (gradient magnitude here); the *CAF* (an inversion); the new defined penalty (the inverse of the gradient magnitude).

The following list records some features quoted in the *Live-Wire* [49] and *Intelligent Scissors* [127] papers:

- Gradient magnitude,
- Direction of the gradient magnitude,
- Laplacian,
- Intensity on the positive (inside) side of the boundary,
- Intensity on the negative (outside) side of the boundary,
- Intensity on the boundary (edge).

The cost assignment process depends on how one wants to emphasis one or another value of the feature. For the gradient magnitude the inverse can for example be taken as a *CAF* in order to favor high contrasts, but a Gaussian function, centered on the gradient value one wants to highlight, could also be applied. For the Laplacian feature, the *CAF* is usually a zero-crossing detector. But many other functions may be used according to the feature values to highlight.

Once satisfying individual cost functions are available, the last step consists in combining them into a total cost function. Let us call potential the weighted sum

of all the individual cost functions. This word of potential comes from the Active Contours approach where the energy of the boundary is defined as the integral along this boundary of a functional called potential. On the directed graph-arc from a pixel p to an adjacent pixel q, the potential used by *Intelligent Scissors* [127] is defined by equation:

$$\begin{aligned} \mathcal{P}(p,q) &= \omega_g \mathcal{C}_g(q) + \omega_L \mathcal{C}_L(q) + \omega_d \mathcal{C}_d(p,q) \\ &+ \omega_i \mathcal{C}_i(q) + \omega_o \mathcal{C}_o(q) + \omega_e \mathcal{C}_e(q) \end{aligned} \tag{3.3}$$

where $\mathcal{C}_x$ are the cost functions associated to the features as follows:

- $\mathcal{C}_g$: gradient feature;
- $\mathcal{C}_d$: gradient direction feature;
- $\mathcal{C}_L$: Laplacian feature;
- $\mathcal{C}_I$: inside intensity feature;
- $\mathcal{C}_O$: outside intensity feature;
- $\mathcal{C}_E$: edge intensity feature;

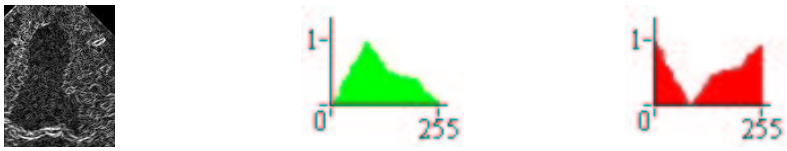
and each ω_x is the weight of the corresponding cost function.

The energy of a path is then defined by

$$E_{\text{path}} = \sum_{(p,q)\in \text{path}} \mathcal{P}(p,q) \tag{3.4}$$

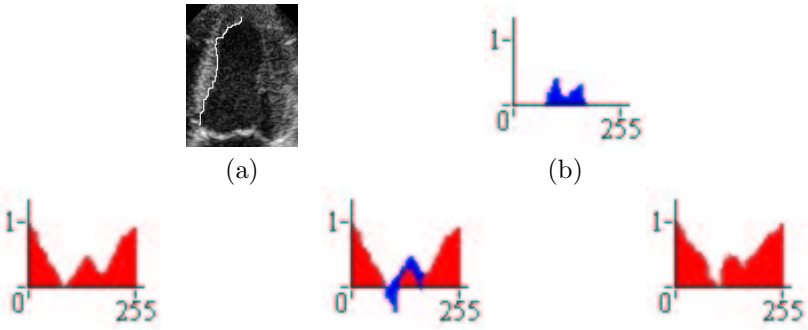
On-The-Fly training

For some features it is hard to decide without prior knowledge about the boundary to extract which values are to be preferred in the cost function. The notion of *On-The-Fly* training is introduced in *Intelligent Scissors* [127] and consists in adapting, during the extraction, the cost functions of the features to the specificities of the contour one wants to segment. It is done for each feature independently from each other. The idea is the following: assuming that the user has drawn a long enough and valid boundary segment, the cost function of the feature has to be modified in order to favor contours with the same feature-values than those found on the segment. An example with the edge intensity feature is shown in table: if the extracted boundary segment is rather dark, the cost function will be modified in order to favor dark intensities. In practice, the process does not modify directly the cost function but the *CAF* : the feature values found on the valid boundary segment (called training path) form an histogram, called training histogram, and the cost *CAF* is iteratively modified by removing from it the training histogram and scaling the result between 0 and 1. Figure 3.24 and 3.25 illustrates respectively the initialization and an iteration of this process. The interest of *On-The-Fly* training is that the potential can be adapted during the extraction, providing the possibility of following a contour with slowly changing properties.



(a) (b) (c)

Figure 3.24. Training initialization: (a) the features trained; (b) its corresponding histogram at initialization; (c) its corresponding cost assignment function.



(a) (b) (c) (d) (e)

Figure 3.25. Training iteration: (a) the trained path; (b) the histogram along the trained path; (c) the cost assignment function at iteration i; (d) the CAF minus the trained histogram; (e) the scaled CAF for iteration $i + 1$.

Path-Cooling

With the minimal path approach it is impossible to extract a closed contour with only one seed point and one end point. Indeed, it would mean that two different paths could pass through a same point. To extract a close contour two seed points, at least, are needed. Furthermore, if the extracted path becomes too long, the path search method will prefer shorter paths cutting through the background: it is often necessary to fix several points to draw the expected contour. *Path-Cooling* was introduced in *Intelligent Scissors* [10], as *Bordery Cooling* and achieves automatic generation of seed points. When a new seed point is generated, the boundary segment between this new point and the previous seed point is fixed (frozen). A new start point is generated when a pixel in the contour is considered to be stable enough. The stability criterion is function of both the time spent on the active boundary segment and the path coalescence (in other terms: how many times a point has been "drawn"). For every pixel in the image two counts are considered: the time history (in milliseconds) counts how long the pixel has been included in the active boundary (boundary section that is not frozen), and the redraw history counts how many times the pixel has been redrawn by the active boundary. When the mouse moves, drawing a new optimal path, the redraw history of the active pixels is incremented while the redraw history of the non-active pixels is set to 0, and the time history of the active pixels is updated

by adding to the current value the while that the boundary segment was displayed and a gradient term (to have a link with the data).

Both histories have two thresholds: a low and a high. When a pixel in the active boundary satisfies the two low thresholds it becomes a candidate point. The first candidate pixel which active boundary segment contains a pixel that satisfies the two high thresholds becomes the new seed point and the rest of the active boundary is frozen. The low thresholds have to be small in order to select candidates as close to the current free point as possible. The high thresholds have to be relatively large to freeze relatively long segments.

3.2.2 Adaptations and improvements

Our work is more based on the *Intelligent Scissors* than on the *Live-Wire*. In this way, we adopt the pixel based graph version: the nodes of the graph are the pixels, and the oriented arcs represent the oriented links between pixels. Our cost functions are directly inspired from the article [127] and we also use *Path-Cooling* and training. The original contribution essentially lies in the introduction of a more general path search, in the adaptation of the cooling speed, and in the way the training is achieved.

Path search algorithms

One of our tasks was to examine the possibility of using the path extraction detailed in section 1.1.2 in the framework of 2D-*Live-Wire* and *Intelligent Scissors*. This extraction method is based on *Cohen and Kimmel* work [34] and uses *Eikonal equation* for propagation. In our implementation we use several path search methods. We used a modified version of *Dijkstra*'s algorithm, which is the basis of the methods developed in *Live-Wire* [49] and *Intelligent Scissors* [127]. We also developed another implementation based on the path search algorithm proposed by *Cohen and Kimmel* [34], already used for virtual endoscopy in the preceding section. We compare the results obtained with both methods.

Modified version of Dijkstra's algorithm On elongated objects, Dijkstra's classical algorithm might perform poorly. This is due to the minimum cumulative cost principle: as the cumulative cost is defined as a sum along the path, the path's cost is a strictly increasing function of the path's length. Thus, the longer the object is, the more likely will the extraction select shorter paths cutting through the background. See an example in figure 3.26. To overcome this problem, we can use a different expression of the cumulative cost in the Dijkstra's algorithm (see [54]), which allows longer paths, by means of introducing recursivity in the cumulative costs computation (see section 2.5 for details).

Eikonal formulation The main idea of *Cohen and Kimmel* [34] is that the potential and the graph are considered to be continuous, producing a sub-pixel path. With this approach, detailed in chapter 1, the energy to minimize is defined as the integral of a strictly positive functional having low values close to the desired features (see equation (1.3)).

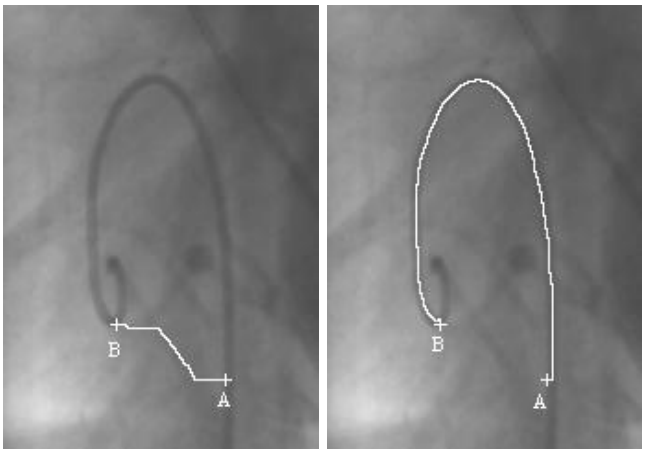


Figure 3.26. Guide-wire extraction in a X-Ray image: Left image - Path extraction with the classical Dijkstra's algorithm between points A and B; right image - Same path extraction with improved Dijkstra's algorithm.

Comparison between Dijkstra and *Eikonal equation*

As explained in the first chapter, the main difference between *Dijkstra* and *Cohen and Kimmel* definition of the minimal path lies in the considered metric. In the first case, the minimal path is the one where the sum of the potential is minimal (L_1 path), and in the second case, it is the one where the integration of the potential is minimal (L_2 path). With Dijkstra's approach, the image is considered as a graph in which each pixel is a node and the weights on the vertices are functions of the energy to minimize. This method uses dynamic programming to compute the optimal path [43]. *Cohen and Kimmel* approach keeps a continuous framework for the problem and computes the path by solving the Eikonal propagation equation in real-time with a Fast Marching algorithm. The aim of the presented work is to achieve real-time extraction. Even if Eikonal method uses integrals to compute the optimal contour, it is not very slower than Dijkstra's approach that uses sums. For the guide-wire extraction shown in figure 3.26, the computing time ration between both methods is about 0.89. And, because *Eikonal equation* produces a sub-pixel path (L_2 path), it is more accurate. The continuous formulation of *Cohen and Kimmel* [34] method has the advantage to keep a more general framework for the energy definition, allowing for example applications using other kind of potentials. It also easily include an offset term w (see equation (1.3)) to constrain the regularity of the path, while it is more difficult with dynamic programming methods [117, 62].

3.2.3 Dedicated potential

To build the potential (i.e the weighted sum of cost functions), two cases are distinguished: the object to extract is either an interface between two regions, or a line (ridge) over a uniform background. The line approach was motivated by the

patents [55, 56], which are based on a ridge filter to proceed to the extraction of linear structures. It was therefore possible to compare quickly the two path search methods (discrete and continuous), and have a rapid overlook over the facilities of the *Live-Wire* and *Intelligent Scissors* tools. For the region interface approach, we use the six features quoted previously: the gradient magnitude, the Laplacian zero crossing, the gradient direction, the edge intensity, the inside intensity and the outside intensity, as described in [127]. Thus, the potential at an oriented arc (p,q) is described by the equation (3.3). In the next section, the defined costs functions have values scaled between 0 and 1. To display the cost maps we re-scale the values between 0 and 255 in this way: if C is the cost function and MC is the cost map associated to C, at a pixel (i,j).

Specific case: long curve extraction

A ridge-filter potential is used for line extraction and is based on local contrast estimation, seen as the difference between a tangential and an orthogonal term, relatively to the direction of the line to extract. It is similar to method described in [96]. Discrete and continuous implementations can be found in [109, 60]. For more details, see [54, 55, 56].

General case: Contour extraction

The potential used for the general case of contour extraction is the one introduced in equation (3.3).

- Gradient magnitude: this feature is useful to locate contrasted areas. As a first order derivative operator, it is a representation of the spatial variations of the image. The gradient of an image I(x,y) is defined by: $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$ and taking $G_m = \sqrt{I_x^2 + I_y^2}$ the cost function is given by

$$\mathcal{C}_g(x,y) = \frac{\max_I G_m - G_m(x,y)}{\max_I G_m - \min_I G_m} \; ; \; 0 \leq \mathcal{C}_g \leq 1$$

  The image is convolved with a Gaussian kernel, before computations.

- the Laplacian zero crossing: As a second order derivative operator, the purpose of the Laplacian zero-crossing is edge localization. The Laplacian of an image I is defined by $L(I) = I_{xx} + I_{yy}$ The corresponding cost function to the Laplacian feature is the Laplacian zero-crossing (LZC). In theory it is defined like this: the LZC is 0 where the Laplacian is 0 and 1 everywhere else. But in practice such a LZC does not produce many zero-crossing points. That is why the zero-crossing area is extended to the pixels where the Laplacian changes its sign with a specific gap. The gap has an influence on the strength of the contrast one wants to highlight with the cost function: the deeper the gap is, the stronger the selected contrast will be. See figure 3.27. Therefore, a zero-crossing is defined by two points with Laplacian of opposite signs The point with its Laplacian

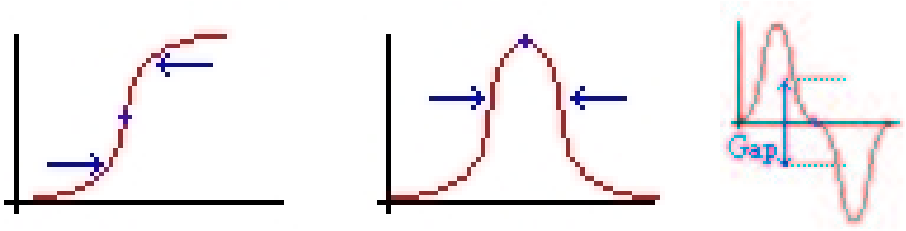


Figure 3.27. Influence of the depth of the gap: left image - profile of an image intensity; middle image - profile of the associated gradient; right image - profile of the associated Laplacian.

closer to zero than the other is set to be the *LZC* . Actually, a pixel is a *LZC* if its Laplacian is zero or:

- First, it is closer to zero than any of its neighbors with a Laplacian from opposite sign,
- Then, there is at least one opposite sign neighbor so that the gap between them is sufficient.

Consequently, the *LZC* map is a binary map defined by:

$$\begin{cases} \mathcal{C}_L(p) = 0, \text{ if } L(p) = 0 \\ \mathcal{C}_L(p) = 0, \text{ if } \exists q \in N(p) \setminus [(L(p).L(q) < 0) \cap (|L(P)| < |L(q)|) \cap (|L(p) - L(q)| < \text{gap})] \\ \mathcal{C}_L(p) = 1 \text{ otherwise} \end{cases}$$

where $N(p)$ is the neighborhood of p.

- The gradient direction: introduces a smoothness constraint into the potential. The associated cost function is not local (i.e. defined on one pixel) but measures the continuity of the gradient direction between two adjacent pixels. We use the formulation as defined in *Intelligent Scissors* [127], for the gradient direction cost from pixel p to pixel q:

$$\mathcal{C}_d(p,q) = \frac{2}{3\pi}\left\{\arccos\left[d_p(p,q)\right] + \arccos\left[d_q(p,q)\right]\right\} \ ; 0 \leq \mathcal{C}_d \leq 1$$

where $\arccos d_p$ and $\arccos d_q$ represent the angles between the link (p,q) and the gradient direction in respectively p and q. d_p and d_q are computed with

$$\begin{aligned} d_p(p,q) &= D'(p).V(p,q) \\ d_q(p,q) &= V(p,q).D'(q) \end{aligned}$$

with

$$\begin{aligned} D'(p) &= \frac{1}{\sqrt{I_x^2 + I_y^2}}(I_y(p), -I_x(p)) \\ V(p,q) &= \frac{1}{\|p-q\|}\begin{cases} q-p, \text{ if } D'(p).(q-p) \geq 0 \\ p-q, \text{ otherwise} \end{cases} \end{aligned}$$

This potential associates a low cost with an edge between two adjacent pixels where the gradient of the pixels and the link between them have similar directions. Furthermore, it associates a high cost with an edge between two adjacent pixels that have similar gradient directions but are almost perpendicular to the link between them. In practice, the efficacy of such a cost is not obvious and we do not use it in the potential. Applying a smoothing operator to the extracted curve after the extraction would probably have a better effect.

- The pixel intensities: The edge intensity is given by the scaled value of the source image at the boundary; the inside intensity is obtained at some offset k from the boundary in the gradient direction and the outside intensity comes from the image intensity at the same offset k from the boundary in the opposite of the gradient direction. Calling $\mathcal{C}_e$, $\mathcal{C}_i$ and $\mathcal{C}_o$ respectively the edge, inside and outside feature costs, their formulations are:

$$\begin{aligned} \mathcal{C}_e(p) &= \frac{1}{255}\mathcal{I}(p) \\ \mathcal{C}_i(p) &= \frac{1}{255}\mathcal{I}(p + k\frac{\nabla\mathcal{I}}{\|\nabla\mathcal{I}\|}(p)) \\ \mathcal{C}_o(p) &= \frac{1}{255}\mathcal{I}(p - k\frac{\nabla\mathcal{I}}{\|\nabla\mathcal{I}\|}(p)) \end{aligned}$$

Without training these features are not used in the potential because it is impossible to decide without prior knowledge about the contour to be extracted which values are to be preferred. The aim of training is to associate with them an adequate *CAF* .

3.2.4 *Path-Cooling* improvement

We tested two different *Path-Cooling* methods. Both are based on the same idea: if a point in the active contour is stable enough, the corresponding boundary segment is frozen. The first process consists in having one counter, the redraw history, and one threshold. The other approach uses the two counters described in [127], with an adaptation: the time history is updated by multiplying (and not adding) a scaled potential-driven factor instead of a simple gradient-driven factor. The multiplication has a weighting effect which gives more influence to pixels with a low potential. At a pixel p and an iteration i of the cooling process, the time history $\mathcal{TH}_i$ is defined as

$$\mathcal{TH}_i(p) = \mathcal{TH}_{i-1} + t_{i-1} \times [1 - \mathcal{P}(p)]$$

where t_i is the consecutive time the path was active until iteration i and $\mathcal{P}$ the penalty in equation (3.3).

The time history is useful if we consider that when the user does not move (redraw history fix, but time history incremented), he wants to emphasize the already drawn contour. But, as both thresholds are to be satisfied, even if the user does move very slow, or does not move at all, the active path will freeze slowly. In practice, the definition of the thresholds is not obvious and the main difficulty lies in the

interpretation of the mouse movement, in terms of speed and acceleration. We can either increase the cooling speed with the mouse cursor speed or decrease it. An argument to increase the mouse speed is the following: if the user moves fast it is because there is no real difficulty with the drawing and because he considers the extracted path as valid. But if the cooling speed is proportional to the mouse speed, and if the user has to define little areas where the contour has lots of details, he must fix a manual seed point: in these areas he must go slow and the cooling process will also go slower. An argument to decrease the cooling speed is the following: the areas where the user goes slowly are the areas which are not very well defined, where the path changes quickly its aspect... , deciding then that the slower the mouse moves the quicker the user wants the path to freeze. The problem is that if the user goes too slowly in other areas (hesitating, for example), he may fix false paths. Even if the second option, with practice and taking care to remain close to the boundary, seems to be the most adequate, no choice is totally satisfying in a general case. However, the *Path-Cooling* is a very satisfying tool to extract closed boundaries, as shown in figure 3.28.

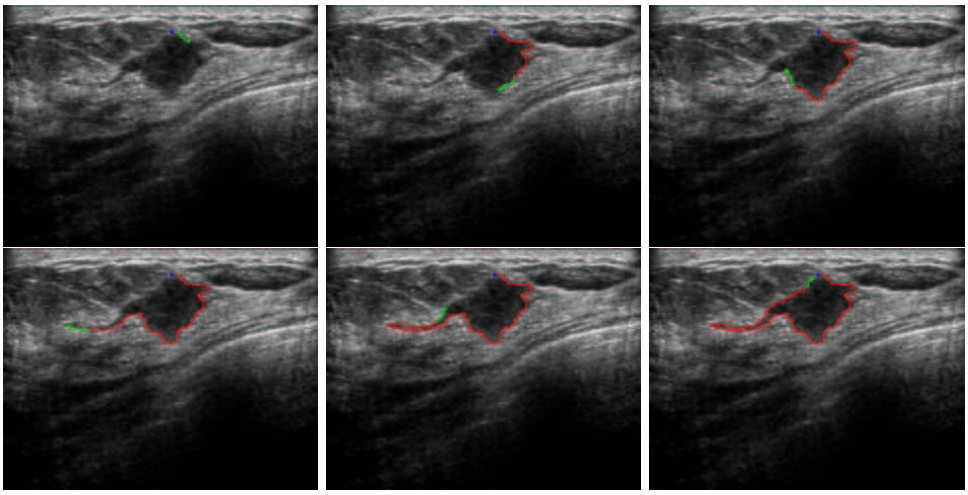


Figure 3.28. *Path-Cooling* on an ultrasound mammography image: These are iterations of the delineation of a tumor; the blue cross is the see point at iteration 0; the green cross is the mouse cursor; the red path is the "cooled" one; the green path is the currently drawn path between the mouse cursor and the seed point at each iteration.

3.2.5 *On-The-Fly* training improvement

In the used potential (see equation (3.3)), training can be applied to the gradient magnitude $\mathcal{C}_g$, inside $\mathcal{C}_i$, outside $\mathcal{C}_o$ and edge features costs $\mathcal{C}_e$.

Training path

Training, as described in [127], is based on the distribution of the feature values on a valid contour segment. The signification of valid is not obvious. We tested three approaches to define such a segment (See figure 3.29):

1. the training path is the last section of the frozen contour and training is applied at each setting of a seed point (blue points in figure 3.29);
2. the training path is the last section of the active boundary and training is applied at each mouse movement, i.e. at each path extraction (green points in figure 3.29);
3. the training path is linked to the cursor position: it corresponds to the points where a mouse movement event is sent to the system, and training is applied at each mouse movement (red points in figure 3.29).

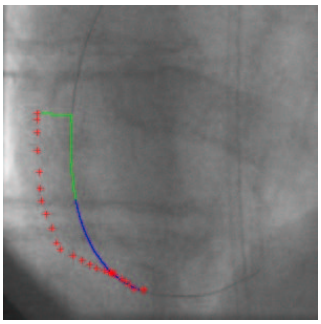


Figure 3.29. Several training paths: On this fluoroscopy image, the cursor position is represented by red crosses, the frozen part of the path is in blue, and the current free part of the path is in green.

Training is effective when a new seed point is manually or automatically set. This setting of a seed point can be seen as the validation of a path segment. The free path, because of its high variability and dependence to the potential (and thus on the training), cannot be a suitable training area. As well as for the mouse movement marker, because it would make it impossible to go too far away from the contour with the mouse cursor.

New way of training

We have developed an improvement of the classical training method (see patent [71]). In existing methods the training area is limited to a portion of the path itself and only takes a positive information into account (see section 3.2.1). The original technique we use is based on the addition of another training area based on negative information. The positive training area contains pixels that could belong to the contour, while the negative training area contains pixels that do not belong to the contour. The positive area has a reinforcing role while the negative area has a penalizing role in

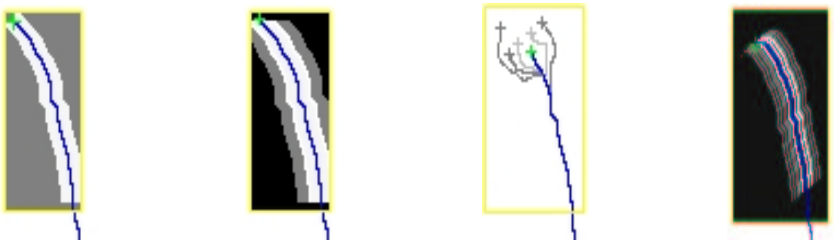
the cost assignment process. This improvement has two consequences: firstly, the new potential is more robust and secondly, the comparison of the distributions of the features (i.e. of the histograms) on each area helps adapting the weights of the individual cost functions in the total potential.

Definition of the positive and negative training areas The main problem is to define suitable positive and negative areas that describe well enough respectively what is and what is not a contour pixel. The definitions we use are all based on the previously described training path. As the drawing of the user is not very accurate, we consider the positive area in the neighborhood of the training path and we tested four approaches for the negative area definition:

1. in the minimal box including the training path, the points closer to the path than a certain distance d are considered to be the positive area, the other points of the box are considered to be the negative area (see figure 3.30-(a));
2. in the minimal box including the training path, the points closer to the path than a certain distance dp are considered to be the positive area and the points further form the path than the distance dp and closer to the path than a certain distance dn are considered to be the negative area (see figure 3.30-(b));
3. the positive and negative areas are made from paths coming from the neighborhood of the click-position. The paths coming from the nearest neighborhood form the positive area and the path coming from the furthest neighborhood form the negative area (see figure 3.30-(c));
4. The training set of points is made from translations of the path: in the minimal box including the training path, the nearest translations are the positive area and the furthest translations are the negative area (see figure 3.30-(d));

For each approach, it is possible to add a weighting function based on the distance to the training path.

The first approach is not accurate enough for the definition of the negative area. Indeed it is possible that other points of the box belong to another right path section. It is the motivation for introducing the second approach. The third method seemed a priori to be the most accurate to define a negative area. But actually, all the paths are rapidly concurrent, what on the first hand misapprehends the notion of positive/negative training point, and on the other hand limits the size of the training set. Second and last approaches are very similar and give the best results with this difference that the first one is a continuous version of the second one. We therefore choose the last approach: the positive area is the set of p nearest translations and the negative area is the set of n next translations of the training path. The translation direction is chosen perpendicular to the mean direction of the training path. The different translations are weighted according to their distance to the training path (see figure 3.31-(d)). The negative/positive areas are symmetric for the gradient magnitude and the edge intensity features (figure 3.31-(a)). But, in order to consider the non symmetrical aspect of the inside and outside features (in equation (3.3)), we adopt for them non symmetric training areas, depending on the direction of the



(a) (b) (c) (d)

Figure 3.30. Different definitions of the positive and negative areas: Positive area in white, negative area in gray, other points of the box in black. (a) Positive area is a lane around the box, negative area is the rest of the box. (b) Positive area is a lane around the box, negative area is a lane around the positive area. (c) Areas built by the paths coming from a region around the mouse cursor. (d) Areas built with translations of the training path.

gradient on the path, the path direction and the considered feature. See examples with figure 3.31-(b)-(c). The positive/negative training sets of points are used to build

(a) (b) (c) (d)

Figure 3.31. Training areas: In white the positive area and in black the negative one. The longest line is the training path. (a) Symmetric. (b) Asymmetric for inside feature. (c) Asymmetric for outside feature. (d) Schematic weighting function.

two distinct histograms of the features values. The function of these histograms is twofold: to build an adapted cost function for the feature (through the construction of an adapted CAF) and to adapt automatically the weight of the individual cost function in the global potential.

***On-The-Fly* adaptation of individual cost functions** Individual cost functions are built with a CAF applied to the feature. Training is used to dynamically modify this cost assignment function. With our method, the algebraic difference between the positive and the negative histograms (jointly scaled) is removed from the iteratively modified CAF and the result is normalized to compose a new cost.

The positive and negative histograms are scaled the following way: firstly, to have a same scale for both training histograms, the negative histogram values are multiplied by the ratio between the number of points used to build the positive histogram and the number of points used to build the negative histogram:

$$H^-[i] = \frac{\text{card}\{H^+\}}{\text{card}\{H^-\}} \times H^-[i]$$

where H^+ and H^- are respectively positive and negative histograms. Then, both histograms are normalized using their common min/max. The initialization used in [127] favors the gray values with highest occurrence in the feature. This is incorrect initialization for images where there is a large and homogeneous background as in figure3.32-(a). We prefer an approach that does not carry any a priori about the expected feature values: the *CAF* is initialized with a flat line, giving the same cost to each pixel of the image, and not with the inverse of the distribution of the feature. As a consequence, the CAF obtained during the process depends only on the past and the present training data.

Computing a cost function using a positive and a negative information into account makes the method more robust. Actually, the positive training area describes what is a contour and the negative training area describes what is not a contour. So if the training contour is not enough uniform, producing a too wide positive histogram, the classical approach will favor points that are perhaps not relevant to the expected contour, whereas our method the negative information to localize the contour, producing a less specific but more accurate cost function.

We show an example of *On-The-Fly* training on the septum wall of an echographic left-ventricle image in figure 3.32.

Figure 3.32 illustrates the first iteration of the process of training on the septum wall of an echographic left-ventricle image (figure 3.32-(a)) with both approaches. The trained feature is the inside intensity one (figure 3.32-(b)). The classical method uses a CAF initialized with the inverse of the distribution of the feature, where black levels are predominant and thus favored (figure 3.32-(c)). The training histogram is scaled between 0 and 1 (figure 3.32-(d)) and removed from the initial CAF to build a new CAF (figure 3.32-(e)) favoring very specific values. With this example the training path is homogeneous and the resulting cost function (figure 3.32-(f)) is good. But with a not uniform enough training contour, the resulting potential will not be consistent. Our method initializes the CAF (figure 3.32-(g)) with an arbitrary value between 0 and 1. The positive and negative histograms (figure 3.32-(h) and figure 3.32-(i)) are jointly scaled and the difference between them is removed from the initial CAF producing a less specific but carrying more accurate information histogram. Thus the new cost function (figure 3.32-(k)) is more relevant.

A real case study of the *On-The-Fly* adaptation of the individual cost functions is shown in figure 3.33, where iterations of the modification of the *CAF* of the feature $\mathcal{C}_e$ are shown.

Adaptation of the weights The principal advantage of using positive and negative training areas is the evaluation of the differences between the histograms, which helps adapting the weight of the corresponding individual cost function in the global potential. If the histograms are enough distinct, we can assume that the considered feature is enough discriminating and that its weight should be more important. In a first approach we take a mean difference between both histograms as dissimilarity criteria, which expression is the following:

$$c = \frac{1}{256} \sum_{i=0}^{255} |H^+[i] - H^-[i]|$$

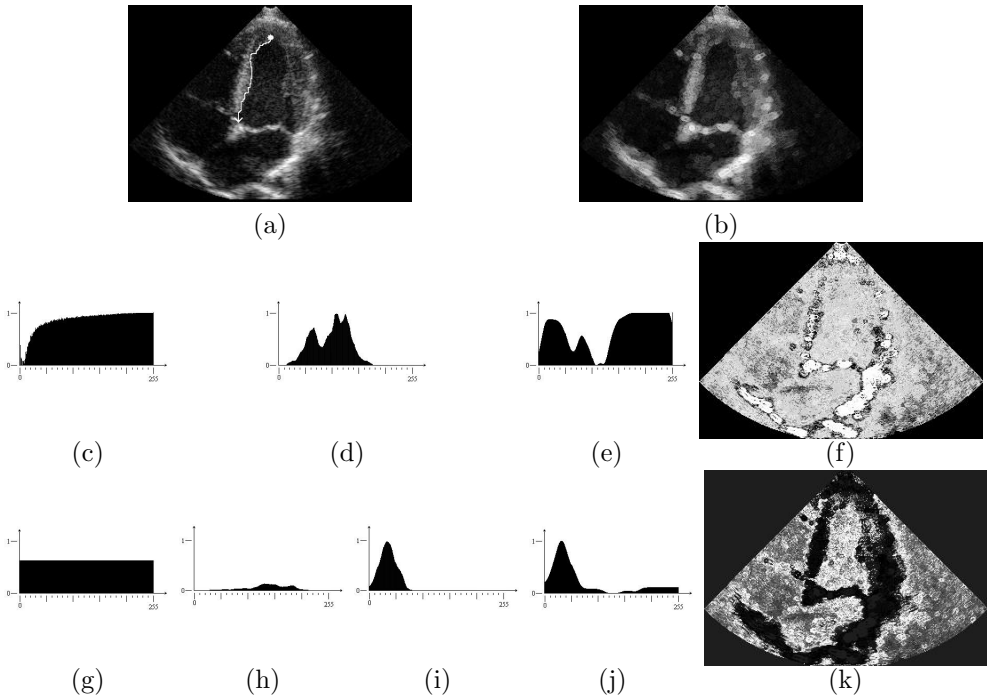


Figure 3.32. Training on the septum wall of an echographic left-ventricle image: (a) echographic left-ventricle image; (b) inside Feature $\mathcal{C}_i$; (c to f) **classical training**: (c) initial CAF ; (d) training histogram; (e) resulting CAF with (f) corresponding cost function; (g to k) **improved training**: (g) initial CAF ; (h) positive and (i) negative training histograms; (j) resulting CAF with (k) corresponding cost function.

Indeed, if the histograms are very similar this criterion will be very low and if they are different, it will be high. In practice, this criterion produces often very low values and quite never values above 0.5. Hence we use a stretching function to exploit efficiently this criterion and transform it into a weight ω_c associated with the cost function c in the total potential. See on figure 3.34 examples of stretching functions.

3.2.6 Conclusion

We have developed an interactive, real-time and user-guided image segmentation software, which gives to a non-specialist the possibility to outline the contour of an object in an image without a very precise drawing (for example with the track-ball on an echograph). Figure 3.35 displays several example of the interactive drawing tool for medical images. Some improvements have been brought to the bibliographical background of interactive extraction of optimal path. A very general optimal path extraction method has been efficiently used, producing in real-time very precise paths. And a new method of *On-The-Fly* training has been developed to adapt, during the

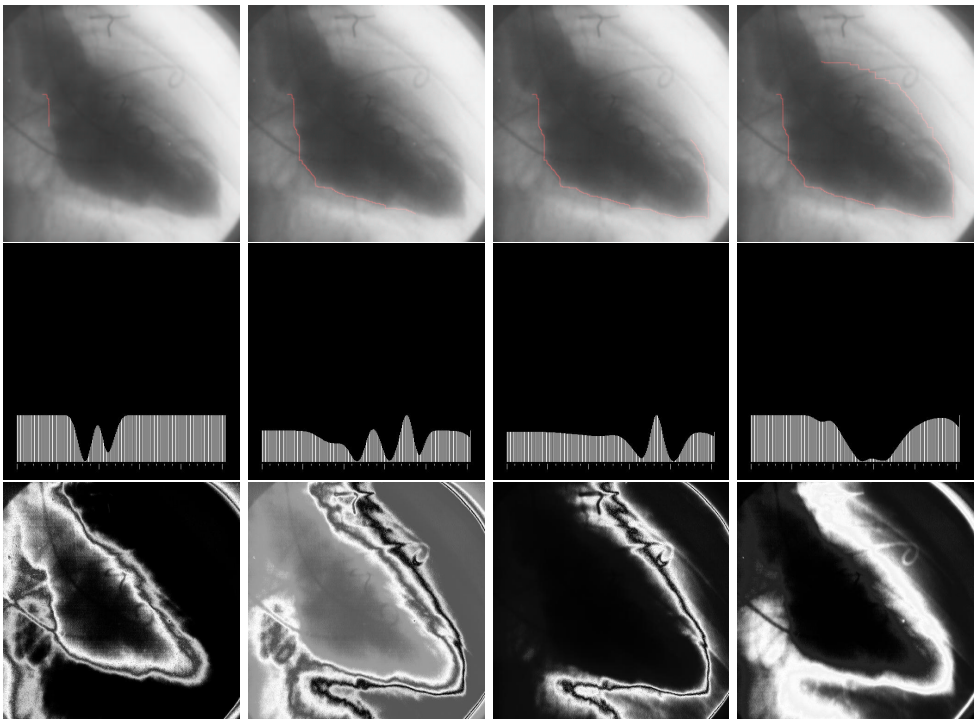


Figure 3.33. Results of the training on a left ventricle image: First row - iterations of the real-time contour extraction with training, on a X-Ray image of the heart left-ventricle; second row - modification of the CAF of the feature $\mathcal{C}_e$ at the same iterations; third row - corresponding feature $\mathcal{C}_e$ potential at the same iterations.

extraction of the contour, the individual cost-functions and their relative weights in the total potential.

3.2.7 Perspective

- Training:
  1. creating a better dissimilarity criterion between information from positive and negative histograms;
  2. ordering the importance of each different computed criteria with the training facility, knowing at each iteration what feature is really discriminant;
  3. use such a method to select *off-line* interesting features into a large database of them (like the size of the Gaussian kernel used to smooth the image).
- *Path-Cooling* : The acceleration of the mouse cursor could be an interesting information for freezing trajectories;

Figure 3.34. Examples of stretching functions: (a) Privileges small values of the cost; (b) penalizes small values of the cost; (c) penalizes values of the cost below a and favors values of the cost above a.

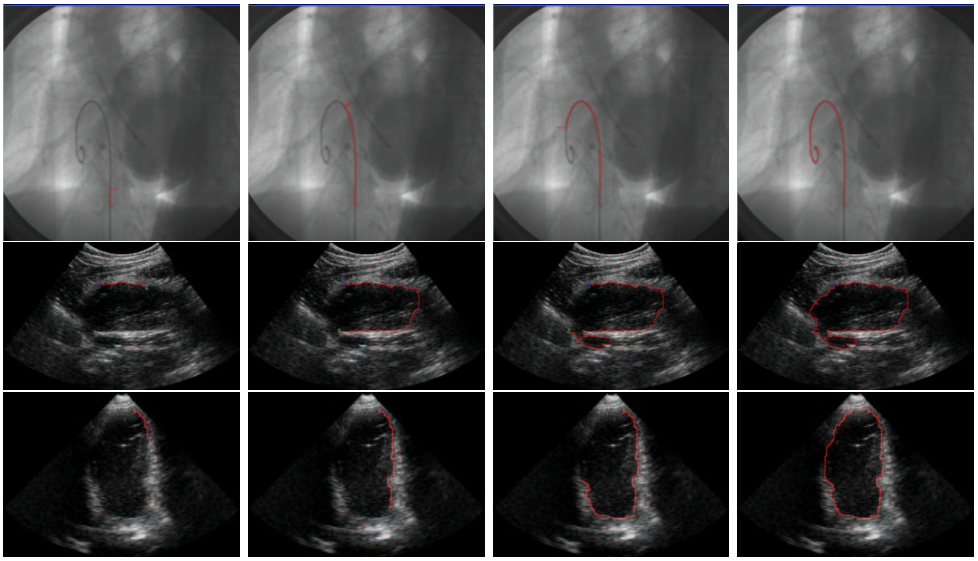


Figure 3.35. Results on different datasets: First row - Extraction of a guidewire in a fluoroscopy image; second row - tumor delineation in a ultrasound mammography image; third row - extraction of the left ventricle in an ultrasound image.

- 3D extension: using a 3D path search method (as in section 2.1), or perhaps a "surface-search", method instead of a slice-by-slice approach ([46, 47]). However, with a slice-by-slice method, a contour extracted with our technique could be an interesting initialization for other pure 3D approaches (as simplex meshes [38] for example).
- Interactivity: grading the level of interactivity, and limit the user interaction to the choice of this level.